PAPER • OPEN ACCESS

Automation of the process of selecting hyperparameters for artificial neural networks for processing retrospective text information

To cite this article: A F Rogachev and E V Melikhova 2020 *IOP Conf. Ser.: Earth Environ. Sci.* **577** 012012

IOP Conf. Series: Earth and Environmental Science **577** (2020) 012012 doi:10.1088/1755-1315/577/1/012012

Automation of the process of selecting hyperparameters for artificial neural networks for processing retrospective text information

A F Rogachev[1,2,*] and E V Melikhova[1]

[1]Volgograd State Agricultural University, 26, Universitetskiy Avenue, Volgograd, 400002, Russian Federation
[2]Volgograd State Technical University, 28, Lenina Avenue, Volgograd, 400005, Russian Federation

*E-mail: rafr@mail.ru

Abstract. Neural network technologies are successfully used in solving problems from various areas of the economy - industry, agriculture, medicine. The problems of substantiating the choice of architecture and hyperparameters of artificial neural networks (ins) aimed at solving various classes of applied problems are caused by the need to improve the quality and speed of deep ins training. Various methods of optimizing ins hyperparameters are known, for example, using genetic algorithms, but this requires writing additional software. To optimize the process of selecting hyperparameters, Google research has developed the KerasTuner Toolkit, which is a user-friendly platform for automated search for optimal hyperparameter combinations. In the described Kerastuner Toolkit, you can use random search, Bayesian optimization, or Hyperband methods. In numerical experiments, 14 hyperparameters varied: the number of blocks of convolutional layers and their forming filters, the type of activation functions, the parameters of the «dropout» regulatory layers, and others. The studied tools demonstrated high optimization efficiency while simultaneously varying more than a dozen parameters of the convolutional network, while the calculation time on the Colaboratory platform for the studied INM architectures was several hours, even with the use of GPU graphics accelerators. For ins focused on processing and recognizing text information in natural language (NLP), the recognition quality has been improved to 83-92%.

1. Introduction

Neural network technologies are successfully used in solving problems from various areas of the economy, including industry, agriculture, and medicine [1, 2]. Monographs and publications in periodicals by F. Scholle, Y. LeCun, Y. Bengio, and G. Hinton [3, 4, 5], as well as Russian researchers S. Nikolenko, A. Kadurina, E. Archangelskaya, I. L. Kashirin, M. V. Demchenko, and A. Sozykin are devoted to substantiating the choice of architecture and hyperparameters of artificial neural networks [6, 7, 8, 9]. We note a number of publications by Jia Y., Kruchinin D., Bahrampour S., devoted to scientific and methodological aspects of ins design and software methods for optimizing their training procedures [10, 11, 12]. The mentioned authors note the problems of justifying the choice of ins architecture and hyperparameters aimed at solving various classes of applied problems. There are known methods for optimizing ins hyperparameters, for example, using genetic algorithms, but this requires writing additional software.

Of particular interest is the publication of L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, dedicated to the Keras Tune tool developed by Google Research to optimize the process of selecting ins hyperparameters [13]. Keras Tuner is an easy-to-use hyperparameter optimization platform that solves problems when searching for a combination of optimal hyperparameters [14, 15]. [15] notes that «... many of today's state-of-the-art results, such as EfficientNet, were discovered via sophisticated hyperparameter optimization algorithms». Currently, this tool is part of the Keras library, but the methodological and applied issues of its application, as well as the effectiveness of various architectures, have not been sufficiently studied. The issues of text data analysis, including in natural language (NLP), are considered in detail by such researchers as B. Bengforth, R. Bilbro, T. Ojeda, H. Palangi, A. Surkova, I. Chernobaev, who note additional difficulties in processing data in Russian [16, 17, 18, 19].

2. Materials and methods

As a convenient software tool for creating software prototypes, the authors used the popular Python V. 3.7 language. To quickly create a software prototype, they used Google Colaboratory, a cloud platform from Google designed to distribute machine learning technologies and deep neural networks. The Colaboratory platform already has a lot of necessary libraries installed, as well as quite powerful Tesla K80 GPUs that significantly accelerate the learning process of neural networks. Kerastuner was used as a tool for searching optimized hyperparameters. It allows creating custom instances of the Hyperband class, the parameters of which are shown in table 1.

Table 1. Functional purpose of arguments in the Hyperband Toolkit.

Name	Appointment of the Argument	Type
hypermodel	Instance of HyperModel class	class
objective	Name of model metric to minimize or maximize	String
max_epochs:.	The maximum number of epochs to train one model	Int
factor	Reduction factor for the number of epochs and number of models for each bracket	Int
hyperband_iterations	The number of times to iterate over the full HyperBand algorithm	Int >= 1
seed	Random seed	Int
hyperparameters	HyperParameters class instance	class
tune_new_entries	Whether hyperparameter entries that are requested by the hypermodel, but that were not specified in hyperparameters should be added to the search space	
allow_new_entries	Whether the HyperModel is allowed to request hyperparameter entries not listed in hyperparameters	

In the Kerastuner Toolkit, you can use Random search, Bayesian optimization, or HyperBand methods [13].

3. Results

In order to study the use of the Keras library's Kerastuner tool on the example of a convolutional ins, software modules for creating a network with hyperparameters that usually do not change during network training were adapted. You must specify a function that will provide variation of the necessary hyperparameters. These parameters were the number of blocks of convolutional layers and their filters, the type of activation functions, parameters of regulatory layers «dropout», types of Pooling, etc. (figure 1). It is possible to set the initial parameter values (default) from the range of variation.

```
import tensorflow as tf
def build_model(hp):
  inputs = tf.keras.Input(shape=(32, 32, 3))
  x = inputs
  for i in range(hp.Int('conv_blocks', 3, 5, default=3)):
    filters = hp.Int('filters_' + str(i), 32, 256, step=32)
    for _ in range(2):
      x = tf.keras.layers.Convolution2D(
        filters, kernel_size=(3, 3), padding='same')(x)
      x = tf.keras.layers.BatchNormalization()(x)
      x = tf.keras.layers.ReLU()(x)
    if hp.Choice('pooling_' + str(i), ['avg', 'max']) == 'max':
      x = tf.keras.layers.MaxPool2D()(x)
    else:
      x = tf.keras.layers.AvgPool2D()(x)
  x = tf.keras.layers.GlobalAvgPool2D()(x)
  x = tf.keras.layers.Dense(
      hp.Int('hidden_size', 30, 100, step=10, default=50),
      activation='relu')(x)
  x = tf.keras.layers.Dropout(
      hp.Float('dropout', 0, 0.5, step=0.1, default=0.5))(x)
  outputs = tf.keras.layers.Dense(10, activation='softmax')(x)

  model = tf.keras.Model(inputs, outputs)
  model.compile(
    optimizer=tf.keras.optimizers.Adam(
      hp.Float('learning_rate', 1e-4, 1e-2, sampling='log')),
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy'])

  return model
```

Figure 1. Creating the architecture of an optimized ANN.

After that, we create an instance of the tuner, which uses the «build_model(hp)» function prepared above for building the model. In the fragment below, the «Hyperband» class of the optimization algorithm will be used to search for ins hyperparameters. Note that you can limit the number of INS launches with the max_trials parameter, which is recommended to be set to the order of several hundred [8].

```
import kerastuner as kt

tuner = kt.Hyperband(
    build_model,
    objective='val_accuracy',
    max_epochs=30,
    hyperband_iterations=2)
```

Figure 2. Building a tuner instance based on the «Hyperband» method.

To test the functioning of the hyperparameter search module, you can use the well-known cifar-10 data set, which is built into TensorFlow. To start the procedure for optimizing the ins parameters, call the «tuner.search» method.

```
import tensorflow_datasets as tfds

data = tfds.load('cifar10')
train_ds, test_ds = data['train'], data['test']

def standardize_record(record):
  return tf.cast(record['image'], tf.float32) / 255., record['label']

train_ds = train_ds.map(standardize_record).cache().batch(64).shuffle(10000)
test_ds = test_ds.map(standardize_record).cache().batch(64)

tuner.search_spase_summary()

tuner.search(train_ds,
             validation_data=test_ds,
             epochs=30,
             callbacks=[tf.keras.callbacks.EarlyStopping(patience=1)])

best_model = tuner.get_best_models(1)[0]
best_hyperparameters = tuner.get_best_hyperparameters(1)[0]
```

Figure 3. The procedure for optimizing the parameters of the Ann using the «tuner.search».

As output values, the module shows the dimension of the search hyperspace and the current values of the variable parameters of the INM, as well as the value «objective». The iterative process of searching for combinations of parameters is quite lengthy and requires the use of a GPU. The optimization software module provides variation of parameters in space with dimension 14. Visualization of the main results of optimization of ins parameters using tuner. search, performed on the Colaboratory platform using GPU graphics accelerators, is shown in figure 4, a)...d).

The diagrams in figure 4 on the ordinate axis show the values of «validate accuracy» achieved by the ins during training on a test sample. Diagram a) represents the influence of the number of convolutional network feature maps on the first layer, diagram b) on the second, diagram c) on the third, and diagram d) represents the influence of the number of neurons in the first hidden convolutional layer.

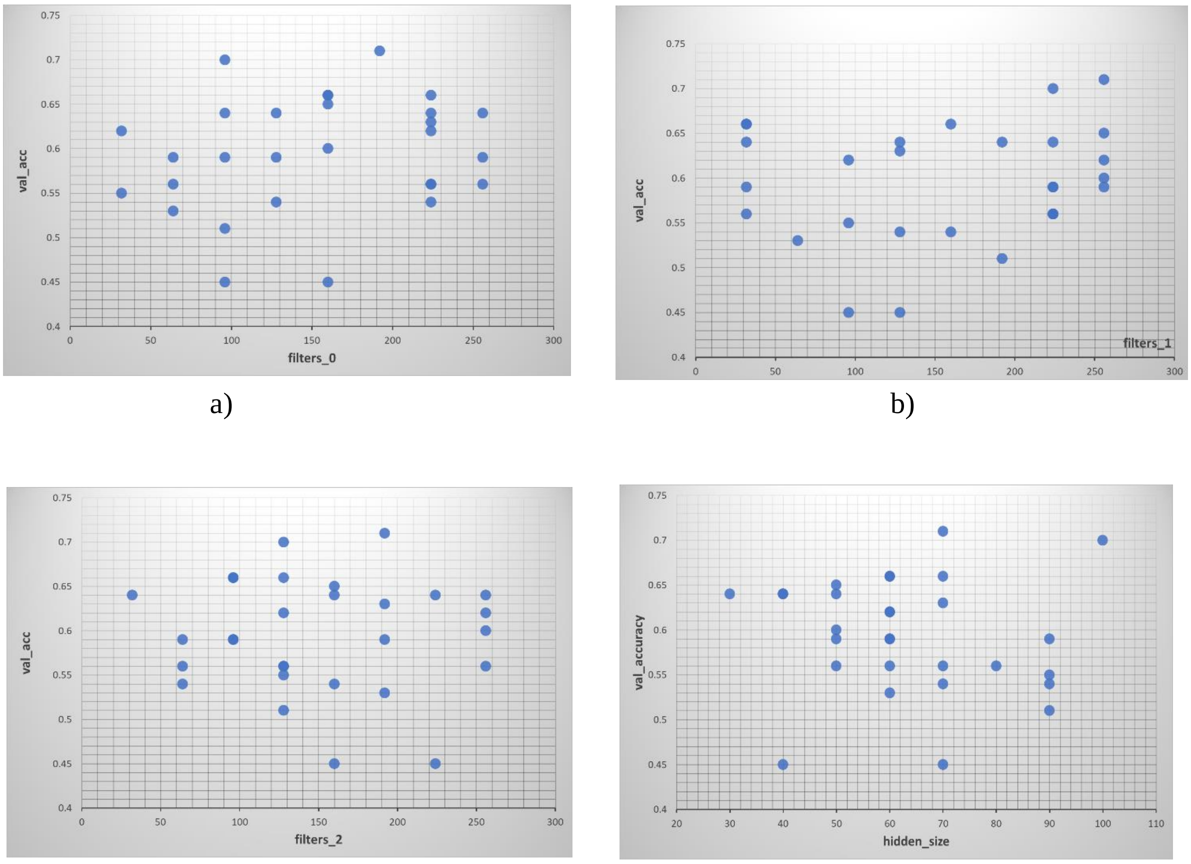
Analysis of the diagram shows the degree of influence of the main hyperparameters of the studied ins on the accuracy of its recognition. The value objective='val_accuracy', calculated from the test sample (figure 3), was taken as an estimated indicator. Each of the variants of the influence of variation of individual hyper-parameters presented in figure 4 is characterized by multimodality,

IOP Conf. Series: Earth and Environmental Science **577** (2020) 012012 doi:10.1088/1755-1315/577/1/012012

especially diagrams a) and b), so it is impossible to unambiguously recommend a priori a combination of preferred values of the studied hyperparameters.

After completing the procedure for selecting a combination of hyperparameters, you can get the best options from the models that were found in the search process, using the «get_best_models» function. It is also possible to view the numerical values of optimal hyperparameters that were found in the automated search process.

Note the significant calculation time, which is several hours even when using GPU graphics accelerators. Optimized neural networks are used to determine the authorship of natural language text corpora prepared for training.



a) b) c) d)

Figure 4. Options for achieving learning accuracy with different combinations of hyper parameters.

Of the key hyperparameters, the number of convolutional layers and the number of neurons in them, as well as the parameters of convolutional layers and their combination, have the greatest influence.

4. Conclusion

The study of the possibility of automated selection of INS hyperparameters using the «kerastuner» tool showed the following.

The «kerastuner» tool demonstrated high optimization efficiency while simultaneously varying one and a half dozen parameters of the convolutional network, but the counting time on the Colaboratory platform for the studied INM architectures was several hours, even with the use of GPU graphics accelerators. For ins focused on processing and recognizing text information in natural language (NLP), the recognition quality has been improved to 83...92%.

Graphical analysis of the influence of variation of individual hyperparameters on the quality of ins operation revealed multimodality of diagrams for various combinations of hyperparameters, especially for the number of feature maps of convolutional layers, so it is impossible to recommend a priori a combination of preferred values of the studied hyperparameters. For this purpose, it is desirable to perform joint automated variation of hyperparameters to improve the quality of the ANN operation.

Acknowledgements
The work was carried out with the financial support of the RFBR and the administration of the Volgograd region under project No. 19-416-340014.

References

[1] Rogachev A and Melikhova E 2019 *IOP Conf. Ser.: Earth Environ. Sci.* **403** 012175
[2] Yann LeCun, Yoshua Bengio and Geoffrey Hinton 2015 Deep learning. *Nature*, **521(7553)**:436–44
[3] Morozov V, Kalnichenko O, Proskurin M and Mezentseva O 2020 Investigation of forecasting methods of the state of complex it-projects with the use of deep learning neural networks *Advances in Intelligent Systems and Computing* **1020** 261-80
[4] Gevorkyan M, Demidova V, Demidova T and Sobolev A 2019 Review and comparative analysis of machine learning libraries for machine learning Discrete and Continuous Models *Applied Computational Science* **27(4)** 305-15
[5] Tahmassebi A 2018 IDEEPLE: deep learning in a flash Proceedings of SPIE - The International Socie 106520S
[6] Tutubalina E and Nikolenko S 2017 Combination of deep recurrent neural networks and conditional random fields for extracting adverse drug reactions from user reviews *Journal of Healthcare Engineering* 9451342
[7] Kashirina I et al 2019 *J. Phys.: Conf. Ser.* **1203** 012090.
[8] Shaikhislamov D, Sozykin A and Voevodin V. 2019 Survey on software tools that implement deep learning algorithms on intel/x86 and Ibm/Power8/Power9 platforms Supercomputing *Frontiers and Innovations* **6(4)** 57-83
[9] Sozykin A et al 2015 Teaching heart modeling and simulation on parallel computing systems *Lecture Notes in Computer Science* **9523** 102-113
[10] Jia Y, Shelhamer E and Donahue J 2014 Caffe: Convolutional Architecture for Fast Feature Embedding *Proceedings of the 22nd ACM International Conference on Multimedia* 675–678
[11] Kruchinin D, Dolotov E and Kornyakov K 2015 Comparison of Deep Learning Libraries on the Problem of Handwritten Digit Classification Analysis of Images, Social Networks and Texts *Communications in Computer and Information Science* **542** 399–411
[12] Bahrampour S, Ramakrishnan N and Schott L Comparative Study of Deep Learning Software Frameworks. https://arxiv.org/abs/1511.06435
[13] Li L and Jamieson K 2018 Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization *Journal of Machine Learning Research* **18** 1-52
[14] Glushchenko A and Petrov V 2019 On comparative evaluation of effectiveness of neural network and fuzzy logic based adjusters of speed controller for rolling mill drive Studies in *Computational Intelligence* **799** 144-50
[15] O'Malley T. 2020 Hyperparameter tuning with Keras Tuner

https://blog.tensorflow.org/2020/01/hyperparameter-tuning-with-keras-tuner.html

[16] Puchkov A, Dli M and Kireyenkova M 2020 Fuzzy classification on the base of convolutional neural networks *Advances in Intelligent Systems and Computing* **902** 379-91

[17] Palangi H et al. 2016 Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP) **24(4)** 694-707

[18] Suvajit D et al 2017 A comparative study of deep learning models for medical image classification *IOP Conf. Ser.: Mater. Sci. Eng.* **263** 042097

[19] Lomakina L, Rodionov V and Surkova A 2014 Hierarchical clustering of text documents *Automation and Remote Control* **75(7)** 1309-15

[20] Zhevnerchuk D, Surkova A, Lomakina L and Golubev A 2018 Semantic modeling and structural synthesis of onboard electronics protection means as open information system *Journal of Physics: Conference Series* **1015** 032157